THE SERIES 1 INTERFACE PLUG-IN FOR TIMEX VERSION OF THE WARAJEVO EMULATOR

This package contains a driver called TSMDR.COM created by Zeljko Juric (with a source file TSMDR.ASM) and a DCK Timex expansion file called SERIES1.DCK created by Keith Watson. These files allow the emulation of the Series 1 Interface Microdrives in Timex mode of the Warajevo 2.5 emulator. As you probably know, the Series 1 Interface is a modification of the normal ZX Interface 1 adapted for use with the Timex Sinclair 2068, which normally does not support this device (this modification is also by Keith Watson). The Series 1 Interface is familiar to Timex users because of a modification to the ROMS.BIN file, supplied in Gerton Lunter's Z80 emulator, which allows partial emulation of the Timex Sinclair 2068 with the Series 1 Interface. Warajevo 2.5 allows full emulation of the Timex, but unfortunately without the Series 1 Interface. This plug-in is dedicated to correct this drawback.

The Timex part of the Warajevo 2.5 emulator will not normally allow the emulation of the microdrive ports because the real Timex had no such device. Fortunately, the Warajevo emulator has the ability to add unimplemented I/O devices using external drivers. If you want details, read more about this in chapter 5 of the Warajevo manual. To prepare for the microdrive emulation, load the TSMDR.COM driver BEFORE starting the emulator and the microdrive ports will be available for use in the Timex emulation. Source code (fully commented) is supported for users that want to learn more about making drivers for unsupported I/O devices.

Microdrive ports without supporting software are not very useful. You will need the ROM code of the Series 1 Interface with system software to use the additional commands for the emulated microdrives. To use the modified ROMS, start the emulator and load the file SERIES1.DCK containing both the modified Timex and Series 1 ROM's as in the example below. As was pointed out before, make sure to load the TSMDR.COM driver first (but, please, don't load this driver twice). Read the Warajevo manual for more details about selecting and starting the Timex part of the emulation. The simplest way to run the Timex part of the Warajevo emulator with the SERIES1.DCK file is is to type the following command from the command prompt (assuming that file SERIES1.DCK is located in the same directory as the emulator itself):

```
TS2068 /ESERIES1
```

You can use the SERIES1.DCK file even if TSMDR.COM is not loaded, but in this case the expanded ROM will be nearly useless.

The Timex and Series 1 ROM's, packed in the file SERIES1.DCK, have been modified to make the microdrives work with the Series 1 code in the DOCK bank. The microdrive files are the same type as and are compatible with Gerton Lunter's MDR files. The only stipulation is that the MDR files for use in this emulation need to be named DRIVE1.MDR, DRIVE2.MDR, and so on up to DRIVE8.MDR (DRIVE1.MDR for microdrive 1, etc.). The user can not change the names of microdrive files, they are fixed. The reason for this is that the TSMDR.COM driver must use as little memory as possible. Keeping eight user-supported filenames for the microdrive files would use additional RAM that is needed more in other areas. If you want only microdrive 1, there is no need for all 8 MDR files to exist. The files DRIVE2.MDR through DRIVE8.MDR need not be present.

The one big difference already mentioned between this emulation and that of the Series 1 microdrives supported with Keith's extension of Lunter's Z80 emulator is that the code for this Series 1 ROM is located in chunk zero of the Timex DOCK bank. A real Series 1 Interface does not use the DOCK bank. It uses a completely independent memory space. The DOCK bank in this emulation is paged into memory in a different way than in the Series 1 ROM used in the Spectrum. In normal use, this page in method is invisible to the user and is of no concern. However, this version of the SERIES1.DCK file does not support the Series 1 code with Zebra's OS-64 LROS program. That's because OS-64 and this Series 1 emulation both use the same chunk zero in the DOCK bank. The size of a Timex memory chunk is 8k. For more information about memory chunks, read chapter 7.9.4. DCK FILES in the Warajevo 2.5 manual.

Another difference is the ability of the Timex to have more than one bank of memory. Since the Timex has more than one bank of native memory, it is possible to LOAD and SAVE programs in the HOME and DOCK banks with this emulation of the Series 1. It is even possible to have a mix of HOME and DOCK bank chunks to SAVE and LOAD. If you want to SAVE memory from mixed chunk specifications, you must be careful when loading a previously saved file because the chunk specification will not be saved with the file. You are on your own when loading a file back to its original location. For example, it is possible to save a DOCK bank binary file, 100 bytes long at address 53744, and later, mistakenly load it back to address 53744 in the HOME bank. The Series 1 code will not load the file back to the same memory chunk specification unless you first write the correct value to port 244. To learn what the memory chunk specification is before a save is performed, you should do: PRINT IN 244. This will return the value that has been sent out to port 244. Record this value, and, before you load the file, do: OUT 244,x, with x equal to the value that was read in from port 244 at the time of the file save. If you plan to use only HOME bank memory, you don't need to be concerned with this information. However, this is a good way to save AROS type programs and load them quickly.

The following math is an example of how to page in the fifth, sixth, and seventh chunks in the DOCK or XROM banks depending on the value written to port 255. DOCK bank memory is enabled when the value written to port 255 is less than 128. XROM bank memory is enabled if the value written to port 255 is greater than 127. For the purpose of determining the paged in chunks, the chunks are numbered zero through seven.

```
OUT 244,(2^4+2^5+2^6)  or  OUT 244,112
```

You should not page in chunk's 0, 1, 2, or 3 of the DOCK or XROM banks when using BASIC commands. If you do, your program will likely crash. The BASIC interpreter is in the first 16k of the HOME bank. The video memory and the variables are located in the third chunk of the HOME bank and the stack is located in the forth chunk of the HOME bank. The only safe way to page in these chunks is with machine code, and they must be paged out of memory before returning control to BASIC.

This emulation cannot save, nor load programs directly from or to the XROM bank. If you have a need for this, use the HOME bank as a temporary buffer and then transfer the memory to the XROM bank by machine code. These comments refer to binary type files only. Although it may possible to setup the top half of the XROM bank to contain BASIC in a real Timex, the modified Timex ROM requires the DOCK bank to be enabled during every error report. Therefore, it is not possible to have BASIC memory existing in the XROM bank in this emulation. The DOCK bank should always be enabled, but not necessarily paged into memory, whenever a BASIC program is running or is in the EDIT process.

The SERIES1.DCK file is configured to have ROM type memory in the first 16K of the HOME bank and RAM type memory in the remaining 48K HOME bank address space (as in a real Timex). The DOCK bank memory is configured for 8K of ROM in the first chunk of DOCK bank memory and RAM type memory in the remaining 56K DOCK bank address space. If you require something different, please read chapter 7.9.4. DCK FILES in the manual that was supplied with the Warajevo 2.5 emulator. It will give you information on the BANK and READ/WRITE header specifications in a DCK files. If you want the Timex or Spectrum ROM space to be configured as RAM, remember that both the Timex and Spectrum ROM's have a bug that will cause parts of the ROM's to be over-written in certain EDIT operations. To prevent this from happening with the Timex ROM, change the byte at address 2151 decimal from 24 to 23 decimal. The same change should be made in the Spectrum ROM code at address 3372 decimal.

The file SERIES1.DCK is a combination of two DCK files combined together to form one file. The first file is the Timex HOME bank ROM with a 9 byte header at the beginning (to describe memory configuration). The second file is the Series 1 ROM with a similar 9 byte header at the beginning. These files are the result of a conversion done in the Warajevo emulator. To convert your own files, please read chapter 3.5.2 SUBMENU 'CONVERT' in the manual. The resulting DCK files were concatenated together in DOS mode as in the following example:

```
COPY /B TIMEX.DCK+SER_1.DCK /B SERIES1.DCK
```

There are some other differences between the ZX Spectrum and Timex Sinclair 2068 Series 1 systems beside the obvious calls to the main ROM. The Spectrum ZX Interface 1 code will add 58 extended variables just below the start of BASIC. This causes the start of BASIC to move up to accommodate these variables. Contrasting with this, this Timex emulation will not cause the start of BASIC to move up. The extended variables will be created below the start of the second display file area and not interfere with programs requiring BASIC to start at address 26710. The Series 1 code does, however, move BASIC up temporarily when accessing the microdrives. A 595 byte buffer is created below BASIC, but, is reclaimed immediately after most microdrive operations. The only thing the user must be careful of is not to overwrite any part of the extended variables with the user's own code before accessing the microdrives. There is one byte used to verify the existence of the extended variables. The Series 1 code will check this byte for a value of 201 at addess 23771 before accessing the microdrives. If it doesn't exist, the extended variables will be recreated, and possibly overwriting the user's own code. These Timex extended variables start at 23760 and are 57 bytes long.

There is one other item of note for users of the ZX Spectrum. The FLAGS3 variable is not located with the other extended variables. Instead, FLAGS3 is located at address 23732, the same as the low byte for the Physical Ramtop variable. FLAGS3 is available for access only during the time the Series 1 ROM is paged in. At any other time, the value will be maintained at 255. The reason is simple but complicated to explain. Simply put, there weren't enough spare variable locations within the IY registers' relative jump range. Instead address 23732 does double duty for both the Physical Ramtop and FLAGS3 variable. There shouldn't be any problem unless you get an error report of ″Ramtop no good″. Even so, it should rarely occur. If it does, execute the command again. If that also fails, try doing the following:

```
POKE 23732,255: POKE 23733,255
```

Try your command once more time. If this error report does happen, the Series 1 ROM did not page out of memory in the normal manner...

Most European users of the ZX Spectrum are familiar with the microdrive syntax. However, very few of the USA users know this syntax. For those that do not know the syntax, refer to the following list of commands (not fully complete, but useful for most users):

TIMEX SERIES 1 SYNTAX	COMMENTS
FORMAT ″m″;n;″name″	This must be done to create cartridge.
CAT n	n = microdrive number, 1 through 8.
LOAD *″m″;n;″file″	″m″ = microdrive channel.
LOAD *″m″;n;″file″CODE	Load a binary file.
LOAD *″m″;n;″file″CODE x	Load a binary file starting at address x.
LOAD *″m″;n;″file″SCREEN$	Load SCREEN file.
SAVE *″m″;n;″file″LINE x	Auto starting program at line x.
SAVE *″m″;n;″file″CODE x,y	Save y bytes at address x.
SAVE *″m″;n;″file″SCREEN$	Save SCREEN file, primary display only.
SAVE *″m″;n;″file″CODE 24576,6912	Save second display file, if enabled.
MOVE ″m″;n;″file″ TO #x	Move data file to stream x.
MOVE ″m″;n;″file″ TO ″m″;n;″file2″	Copy data file to file2.
ERASE ″m″;n;″filename″	Obvious, but don't add CODE for bin files!

VERIFY *"m";n;"filename"	Verify RAM contents match mircodrive's.
MERGE *"m";n;"filename"	Works on HOME bank memory only!!!
OPEN #n;"m";x;"file"	Open a data file for reading or writing.
CAT #n;x	Catalog of Microdrive x sent to stream n.
CLS #	Restore the normal BORDER, PAPER, and INK.
CLEAR #	Close all streams and reclaim memory.
CLOSE #n	Close stream n; reclaim memory used by it.
CLEAR #n	For Timex only. Does the same as CLOSE #n.
INPUT #n;...	Read string in from stream n.
INKEY$ #n	Read character in from stream n.
PRINT #n;...	Output print sequence (...) to stream n.

NOTE: PRINT #n is a normal Timex command which is the same as LPRINT when used as in PRINT #3. Don't use OPEN #3;"t" to print with in this emulation because is sends data to a non-supported serial port.

Other Series 1 commands exist, but they are not functional in this emulation due to non-supported I/O ports, like:

FORMAT "n";x	Set the network station number to x.
FORMAT "t";x or FORMAT "b";x	Set the serial port baud rate.
LOAD *"t" or LOAD *"b"	Commands for the Series 1 serial port.
SAVE *"t" or SAVE *"b"	Commands for the Series 1 serial port.
LOAD *"n";n or LOAD *"n";nCODE	Commands for the Series 1 network.
SAVE *"n";n or SAVE *"n";nCODE x,y	Commands for the Series 1 network.
OPEN #n;"x"	Open binary "b", text "t", or net "n" chan.

NOTE: The separator character in, OPEN #n;"x", must be a ";" in Timex mode even though a "," is allowed in the Spectrum ZX Interface 1. The comma passes the Timex syntax check and does not page in the Series 1 ROM. If used, all you will get is an "Invalid I/O device" error report.